Computer Vision and Image Analysis Course
Lab 1: Camera Calibration and Marker Detection

Interdisciplinary Space Master (ISM)
Interdisciplinary Centre for Security, Reliability and Trust
Computer Vision, Imaging, and Machine Intelligence (CVI^2) group

Course Responsible: Prof. Djamila Aouada
TA: Dr. Arunkumar Rathinam, Dr. Michele Lynn Jamrozik
arunkumar.rathinam@uni.lu; michele.jamrozik@uni.lu

09 November 2022

Introduction

In this lab session, the goal is to perform camera calibration and aruco marker detection using OpenCV and get to know the FLIR camera hardware.

Preliminary -Setup

1. Setup Python Environment

> **Info: Setup Conda Environment**
> » Anaconda installation
> https://docs.anaconda.com/anaconda/install/
> » Miniconda installation
> https://docs.conda.io/projects/conda/en/latest/user-guide/install/index.html
> » Install following packages: opencv, jupyter lab,
> % conda install -c conda-forge opencv
> % conda install -c conda-forge jupyterlab
> % conda install -c conda-forge matplotlib

2. Camera Driver Installation

> **Info:** » install Spinnaker SDK
> https://flir.app.boxcn.net/v/SpinnakerSDK
> » Getting started with spinview
> https://dropit.uni.lu/invitations?share=923a50eea5e199707e8e

3. connect the camera and verify it is working

Task 1: Calibrate Camera

The OpenCV library has a camera calibration toolbox that should be sufficient for most camera calibration needs. You have been provided below with useful instructions to write the code in Python.

1. Connect Camera and collect a set of 10-15 images

   Info: Useful steps
   » collect images using the Spinview GUI
   % open *spinview* application

2. Perform camera calibration

   Info: Useful Opencv Commands
   » read the images using opencv
   % cv.imread(fname)
   » find chess board corners and calibrate camera
   % cv.findChessboardCorners , cv.calibrateCamera

3. Save camera calibration matrix (to use in the report)

Task 2: Marker Generation and Detection

ArUco markers are built into the OpenCV library via the cv2.aruco sub-module. The aruco module in OpenCV has a total of 25 predefined dictionaries of markers. Each dictionary indicates the number of bits and the number of markers contained. Link : OpenCV ArUco Marker Detection

Generate Aruco marker for marker tag **cv.aruco.DICT_6X6_250** and the marker id 60

1. The Opencv-contrib library contains support for Aruco Marker which is not present in the regular opencv installation.

   Info: » To install Opencv-contrib python library.
   $ pip install opencv-contrib-python

2. Use OpenCV to generate ArUco markers

   Info: » get Dictionary information
   % cv2.aruco.Dictionary_get('input_aruco_tag_here')
   » to generate Aruco markers
   % img = cv2.aruco.drawMarker(dictionary, id, sidePixels[, img[, borderBits]])

3. Detect the markers provided in the handout

   Info: » read the images using opencv
   % aDict = cv.aruco.Dictionary_get('input_aruco_tag_here')
   » find chess board corners and calibrate camera
   % aParams = cv2.aruco.DetectorParameters_create()
   % (corners, ids, rejected) = cv2.aruco.detectMarkers(image, arucoDict, parameters=aParams)

4. Print image with ids overlayed with bounding box for each marker.

Task 3: Calibrate a camera using Aruco markers

Use the provided Aruco grid board to calibrate the camera. Use the FLIR camera to acquire image of the provided ARUCO grid board and use the image to compute calibration parameters. Link : OpenCV ArUco

1. Calibrate camera using Aruco gridboard

> **Info:**
>
> » calibrate Camera Aruco opencv
>
> % cv2.aruco.calibrateCameraAruco(corners, ids, counter, board, imageSize, cameraMatrix, distCoeffs[, rvecs[, tvecs[, flags[, criteria]]]]
>
> ?? provide cameraMatrix=None and distCoeffs=None during input
>
> ?? to generate board information
>
> % aruco_dict = aruco.getPredefinedDictionary(aruco.DICT_5X5_50)
>
> % board = aruco.GridBoard_create(markersX=5, markersY=7, markerLength=0.04, markerSeparation=0.01, dictionary=aruco_dict)

2. Save camera calibration matrix (to use in the report)

To Explore: Perform posture Estimation using Aruco marker

Note:

- *Report submission is not needed for this session. But a combined report with results (or Jupyter notebook with proper explanation) is needed at the end of lab session 3 (1-3).*